aa-8. コンピュータにおけるデータ表現とデータエンジニアリング

（人工知能）

URL: https://www.kkaneko.jp/ai/mi/index.html

金子邦彦

この授業でできるようになること

Python 技術スキル

- CSV形式の**表形式データ**を**Pandas データフレーム**として読み込む
- 特定の列を選択し，**条件でデータを絞り込む**
- **欠損値の有無と場所を特定**し，基本的な処理
- **カテゴリ変数を数値データに変換**
- **基本統計量**の算出

基礎の理解、活用力

- ExcelとPythonの使い分けを理解し説明できる
- データの散らばり具合，最頻値付近，極端な値の存在を判断できる
- データ分析やAI開発において倫理的配慮が必要な理由を理解し，述べることができる

前準備

プログラミングの基礎

- **変数の代入**：データや計算結果を変数に格納する方法
- 関数／メソッドの呼び出し：**関数名()、オブジェクト.メソッド()の形式**
- 基本的なデータの種類：**文字列，数値**
- ライブラリのインポート：**Python の import で外部ライブラリをインポート**。機能を拡張

この程度の基礎が、高度なAI、データサイエンスの実践につながる

演習

最初のコードセル

```
# ライブラリの読み込み
import pandas as pd

# 変数の代入（文字列と数値）
student_name = "田中太郎"
student_age = 20

# 関数の呼び出し（組み込み関数）
print("学生名:", student_name)
print("年齢:", student_age)

# 辞書型データの作成（演習用データと同じ構造）
student_data = {
    "名前": ["田中太郎", "佐藤花子", "鈴木次郎"],
    "年齢": [20, 19, 21],
    "身長": [170, 158, 175],
    "体重": [65, 50, 72],
    "学年": [2, 1, 3]
}
```

8-1 Python による表形式データの扱い（Pandasを使用）

ここでの目標

- **CSV形式の表形式データ**をPandas **データフレーム**として**読み込む**
- **データフレームの基本構造**（行数，列数，列名，データ型）を確認
- データフレームから**特定の列を選択**，**簡単な条件**で**データを絞り込む**

コンピュータにおけるデータ表現

アナログからデジタルへ

- アナログ情報：連続的な値（温度，音声，画像など）
- デジタル情報：離散的な値（0と1の組み合わせ）

デジタル化の利点：

- 正確な複製が可能
- 効率的な処理が可能
- ネットワークでの伝送が容易

データの種類

- 数値データ：整数，浮動小数点数
- 文字列データ：テキスト情報
- 論理値データ：True/False
- 日付・時刻データ：時系列情報

CSV形式

- Comma Separated Valuesの略
- 行と列で構成される**表形式データ**
- **各行がデータの1つのレコード（記録）**を表す
- 1行目は通常，列名（ヘッダー）となる

```
名前,年齢,身長,体重,学年
田中太郎,20,170,65,2
佐藤花子,19,158,50,1
鈴木次郎,21,175,72,3
山田美咲,20,162,52,2
高橋健太,22,168,70,3
```

Pandasライブラリの重要性

Pandasデータフレームの特徴

- 高性能：大量データの高速処理
- 多機能：データ処理に必要な機能を網羅
- **データフレームは、表形式のデータ**

Pandasでできること

- データの読み込み・書き出し
- データのクリーニング・前処理
- データの変換・集計
- 欠損値の処理
- データの結合・マージ
- 統計計算
- データの可視化（基本的なもの）

CSVファイルの読み込み

読み込み処理の流れ

- ファイルの読みこみ
- **区切り文字の認識**：カンマ（,）の識別
- **ヘッダー行の処理**：列名の設定
- **データ型の推定**：各列のデータ型を自動判定

pd.read_csv()　Pandas ライブラリ内（別名 pd）の read_csv()関数の呼び出し

→　**上の処理をすべて行う**関数を呼び出す

データフレームの特徴

- **各列のデータ型**：数字（int64, float64）など
  int は整数、float は小数付きの数、64 は 64ビットデータの意味
- **列名**：データの意味を表す名前

データフレームの特徴を得るためのメソッド

- **df.shape**

用途：データサイズの把握

- **df.columns.tolist()**

結果の例：['名前', '年齢', '身長', '体重', '学年']
用途：列名の確認

- **df.dtypes**

結果の例：int64, float64
用途：各列のデータ型の確認

データフレームのデータ選択（列選択とデータ絞り込み）

列選択の種類と用途

- **単一列選択**

構文：df['列名']

- **複数列選択**

構文：df[['列名1', '列名2']]

条件による絞り込み

- 基本構文

df[条件式]

データ選択の実践的応用

ビジネス分析での活用例

- **顧客セグメンテーション**

年齢層別の顧客分析、購買行動パターンの特定

- **売上分析**

特定期間の売上データ抽出、**商品カテゴリ別**の分析

- **品質管理**

基準値を超える製品の特定、**不良品**の原因分析

演習　セル１～４による演習

セル1: インポートと演習用サンプルデータ（CSV形式）の作成・保存

```
=== セル1: CSVファイル作成と基本操作 ===
CSVファイルを作成しました

データの形状: (8, 11)
行数: 8, 列数: 11

列名: ['学生ID', '名前', '年齢', '身長', '体重', '学年', '専攻', '成績評価', '数学', '英語', '物理']

データ型:
学生ID      object
名前        object
年齢       float64
身長         int64
体重       float64
学年         int64
専攻        object
成績評価      object
数学       float64
英語       float64
物理       float64
dtype: object
```

int は整数、float は小数付きの数、

64 は 64ビット

object は数字以外のデータ

セル2: CSVファイルの読み込み

- CSV形式は表形式データを保存する最も基本的な形式の一つである
- CSVファイルは行と列で構成され，各行がデータの1つのレコード（記録）を表す
- PythonのPandasライブラリを使用することで，大量のデータを効率的に処理できる

```
=== セル2: CSVファイルの読み込み ===
CSVファイルを読み込みました
   学生ID     名前    年齢    身長    体重  学年  専攻 成績評価    数学    英語    物理
0  S001   田中太郎  20.0   170   65.0   2  工学    優  85.0  78.0  92.0
1  S002   佐藤花子  19.0   158   50.0   1  文学    良   NaN  82.0   NaN
2  S003   鈴木次郎  21.0   175    NaN   3  工学    優  95.0  89.0  88.0
3  S004   山田美咲  20.0   162   52.0   2  理学    可  72.0   NaN  76.0
4  S005   高橋健太   NaN   168   70.0   3  工学    良  88.0  91.0  85.0
5  S006    伊藤舞  19.0   155   48.0   1  文学    優  76.0  85.0  79.0
6  S007   渡辺大輔  22.0   178   75.0   4  理学    良  82.0   NaN  91.0
7  S008   中村雅子  21.0   160    NaN   2  工学    可   NaN  78.0  83.0
```

セル3: データ構造の確認

【授業知識】データフレームの理解

* DataFrameは表形式のデータ構造で，行（レコード）と列（カラム）で構成される
* 各列には名前（列名）があり，同じ列のデータは同じ種類の情報を表す

```
=== セル3: データ構造の確認 ===
データの形状: (8, 11)
行数: 8, 列数: 11

列名: ['学生ID', '名前', '年齢', '身長', '体重', '学年', '専攻', '成績評価', '数学', '英語', '物理']

データ型:
学生ID      object
名前        object
年齢       float64
身長         int64
体重       float64
学年         int64
専攻        object
成績評価      object
数学       float64
英語       float64
物理       float64
dtype: object

最初の5行:
   学生ID    名前    年齢   身長    体重  学年  専攻 成績評価    数学    英語    物理
0  S001  田中太郎  20.0  170  65.0   2  工学    優  85.0  78.0  92.0
1  S002  佐藤花子  19.0  158  50.0   1  文学    良   NaN  82.0   NaN
2  S003  鈴木次郎  21.0  175   NaN   3  工学    優  95.0  89.0  88.0
3  S004  山田美咲  20.0  162  52.0   2  理学    可  72.0   NaN  76.0
4  S005  高橋健太   NaN  168  70.0   3  工学    良  88.0  91.0  85.0
```

セル4: 列選択とデータの絞り込み

【授業知識】データの選択と絞り込み

* データ分析では，全てのデータではなく特定の列や条件に合うデータのみを扱うことが多い
* 列選択は角括弧[]を使用して行い，複数列の場合はリスト形式で指定する
* 条件による絞り込みは，特定の基準を満たすデータのみを抽出
* この操作により，大量のデータから必要な情報だけを効率的に取り出すことができる

```
=== セル4: 列選択とデータの絞り込み ===
名前と年齢の列を選択:
      名前    年齢
0   田中太郎  20.0
1   佐藤花子  19.0
2   鈴木次郎  21.0
3   山田美咲  20.0
4   高橋健太   NaN
5    伊藤舞  19.0
6   渡辺大輔  22.0
7   中村雅子  21.0

工学専攻の学生のみを絞り込み:
      名前  専攻  学年
0   田中太郎  工学   2
2   鈴木次郎  工学   3
4   高橋健太  工学   3
7   中村雅子  工学   2

年齢が20歳以上の学生:
      名前    年齢
0   田中太郎  20.0
2   鈴木次郎  21.0
3   山田美咲  20.0
6   渡辺大輔  22.0
7   中村雅子  21.0
```

ここまでのまとめ

技術的スキル

- データフレームの基本構造の確認
- 列選択（単一列・複数列）の実行
- 条件によるデータ絞り込み

確かな基礎

- コンピュータにおけるデータ表現、表形式データ
- CSV形式とデータフレームの関係性
- Pandasライブラリの重要性と特徴

休憩とQ&A

よくある質問

- Q: データフレームとExcelの表は同じですか？

A: 基本的な構造は似ていますが，Pythonでは大量データを高速処理でき，自動化も可能です

- Q: インデックスは必ず0から始まりますか？

A:　基本は0からです。（高度な設定により変更可能です）

- Q: エラーが出た場合はどうすれば良いですか？

A: エラーメッセージを読み，スペルミスや括弧の対応を確認しましょう

8-2 欠損値

学習目標

- **データセット内の欠損値の有無と場所**
- **欠損値**に対する**基本的な処理（削除）**をPandasで実行できる
- **文字列のカテゴリ変数を数値データに変換**する**replace関数**

現実のデータの「不完全さ」、データクリーニング

欠損値
•データが入力されていない箇所
- センサーの故障
- 回答拒否・記入漏れ
- システムエラー

データ形式の不統一
文字列と数値の混在
日付形式のばらつき
大文字・小文字の不統一

外れ値・異常値
- 入力ミスによる異常な値
- 測定機器の誤作動

データクリーニングの重要性
- 分析結果の信頼性確保
- AI（機械学習）の性能向上
- 自動化処理の安定性確保
- エラーの事前防止

欠損値処理の戦略

主要な欠損値処理手法

- 削除（Deletion）

行削除：欠損値を含む行を除去
列削除：欠損値の多い列を除去
→　本授業で重点的に学習

他の方法もある

欠損値削除処理

メソッド：df.dropna()
データフレーム全体について、欠損値を含む行を一括除去
　適用場面：欠損値が少数の行に集中

メソッド：df.dropna(axis=1)
データフレーム全体について、欠損値を含む列を一括除去
　適用場面：特定の列に欠損値が集中

カテゴリ変数の変換

カテゴリ変数とは

- **限られた選択肢から成る変数**
- 文字列で表現されることが多い

具体例

性別：「男性」「女性」
評価：「高」「中」「低」
学年：「1年」「2年」「3年」
地域：「東京」「大阪」「名古屋」

数値化の必要性

- 計算処理の効率化
- AI（機械学習）でも必要

AIの仕組み上、数値入力が前提であることが多い

カテゴリ変数変換手法

主要な変換手法

- 置換法（Replace）

本授業で学習

- **文字列→数値変換**
- 実装が簡単で理解しやすい

その他の高度な手法

　自動的な連番割り当てなど

replace関数の特徴と利点

- 変換ルールが明確
- 変換ルールを設定可能

基本的な使用法

df['列名'].replace({'変換前': 変換後, ...})

演習　セル５～７による演習

セル5: 欠損値の確認

【授業知識】欠損値の理解と重要性

- 現実のデータには必ずと言っていいほど欠損値（データが入っていない部分）が存在する
- 欠損値の存在はデータ分析結果に大きな影響を与える可能性がある
- 欠損値を無視して分析を行うと，誤った結論に至る危険性がある
- データクリーニングの第一歩は，欠損値の存在とその場所を正確に把握することである

欠損値の場所を確認:

	学生ID	名前	年齢	身長	体重	学年	専攻	成績評価	数学	英語	物理
0	False	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	True	False	True	
2	False	False	False	False	True	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	True	False
4	False	False	True	False	False	False	False	False	False	False	False
5	False	False	False	False	False	False	False	False	False	False	False
6	False	False	False	False	False	False	False	False	False	True	False
7	False	False	False	False	True	False	False	False	True	False	False

欠損値を含む行:

	学生ID	名前	年齢	身長	体重	学年	専攻	成績評価	数学	英語	物理
1	S002	佐藤花子	19.0	158	50.0	1	文学	良	NaN	82.0	NaN
2	S003	鈴木次郎	21.0	175	NaN	3	工学	優	95.0	89.0	88.0
3	S004	山田美咲	20.0	162	52.0	2	理学	可	72.0	NaN	76.0
4	S005	高橋健太	NaN	168	70.0	3	工学	良	88.0	91.0	85.0
6	S007	渡辺大輔	22.0	178	75.0	4	理学	良	82.0	NaN	91.0
7	S008	中村雅子	21.0	160	NaN	2	工学	可	NaN	78.0	83.0

欠損値確認で使用するメソッド

- df.isnull()

各セルが欠損値かどうかをTrue/Falseで返す

セル6: 欠損値の処理

【授業知識】欠損値処理の基本戦略（削除処理中心）

- 欠損値への対処法として，本授業では「削除」を中心に学習する
- 削除は最も基本的で理解しやすい処理方法である
- 削除処理では，データ量が減少し情報が失われる可能性があることを理解する必要がある
- 削除による影響（失われるデータの量と性質）を必ず確認することが重要である

```
=== セル6：欠損値の処理 ===
元データの行数：8

欠損値を含む行を削除：
削除後の行数：2
削除された行数：6

削除後のデータ：
   学生ID     名前     年齢    身長     体重   学年   専攻 成績評価     数学     英語     物理
0  S001   田中太郎   20.0   170   65.0    2   工学     優   85.0   78.0   92.0
5  S006    伊藤舞   19.0   155   48.0    1   文学     優   76.0   85.0   79.0

特定の列の欠損値のみ削除（数学成績）：
数学成績の欠損値削除後：6行
```

セル7: カテゴリ変数の数値変換

【授業知識】カテゴリ変数の数値化の必要性（基本レベル）

- コンピュータは数値データの方が文字列データよりも効率的に処理できる
- 統計分析や機械学習では，文字列で表現されたカテゴリを数値に変換する必要がある
- 本授業では基本的なreplace関数を用いた変換方法を学習する

```
=== セル7: カテゴリ変数の数値変換 ===
元の成績評価:
成績評価
優    3
良    3
可    2
Name: count, dtype: int64

replace関数を用いた変換:
変換結果:
     名前 成績評価  成績評価_数値
0  田中太郎    優         3
1  佐藤花子    良         2
2  鈴木次郎    優         3
3  山田美咲    可         1
4  高橋健太    良         2
5   伊藤舞    優         3
6  渡辺大輔    良         2
7  中村雅子    可         1
```

専攻の数値変換:

	名前	専攻	専攻_数値
0	田中太郎	工学	1
1	佐藤花子	文学	2
2	鈴木次郎	工学	1
3	山田美咲	理学	3
4	高橋健太	工学	1
5	伊藤舞	文学	2
6	渡辺大輔	理学	3
7	中村雅子	工学	1

ここまでのまとめ

技術的スキル

- 欠損値の有無と場所の特定
- 欠損値削除処理の実行
- カテゴリ変数の数値変換（replace関数）

確かな基礎

- 現実データの「不完全さ」の理解
- 欠損値処理の必要性
- カテゴリ変数数値化の必要性

8-3 基本統計量、ヒストグラム

学習目標

- **基本統計量（平均値，中央値，最大値，最小値）**を，Pythonを用いて効率的に算出
- **ExcelとPythonの使い分け**（処理速度，大量データ対応）を理解

基本統計量

- **平均値**（Mean）：データの算術平均
- **中央値**（Median）：データを順番に並べた時の中央の値
- **最頻値**（Mode）：最も頻繁に現れる値
- **範囲**（Range）：最大値と最小値の差
- 分散（Variance）：平均からの偏差の平方の平均
- **標準偏差**（Standard Deviation）：分散の平方根

目的

- データの全体像を把握
- 異なるグループ間の比較
- 時系列での変化の把握
- 客観的データに基づく判断

ExcelとPythonの統計処理比較

Excel の特徴

- 視覚的で直感的な操作
- 簡単
- プログラミング無しで可能

Python の特徴

- 大量データの高速処理
- 自動化・再現性に優れる
- 高度な統計・機械学習手法も備わる

Python の Pandasでの統計量計算

```
# 基本統計量の一括計算
df.describe()

# 個別の統計量計算
df['列名'].mean()    # 平均値
df['列名'].median()  # 中央値
df['列名'].max()     # 最大値
df['列名'].min()     # 最小値
df['列名'].std()     # 標準偏差
df['列名'].var()     # 分散
df['列名'].count()   # データ数
```

- 一度のコマンドで全統計量算出
- 欠損値の自動除外
- 高速な並列処理

データ可視化の重要性

可視化の効果

- 複雑なデータパターンの把握
- 瞬時の全体像理解
- 異常値・外れ値の発見
- コミュニケーション向上
- 分析の品質向上
- 仮説の検証・発見

ヒストグラムの特徴

- データの度数分布を視覚化
- 区間（bin）ごとの頻度表示

読み取ることができる情報

- データの中心位置
- 散らばり具合
- 分布の形状（正規分布，偏り等）
- モード（最頻値）の位置
- 外れ値の存在

分布形状の判定と意味

主要な分布形状の種類

〇 正規分布（ベル型）

- 左右対称
- 平均値付近にデータが集中
- 多くの統計手法の前提

分布形状の判定方法

- ヒストグラムの形状観察
- 対称性の確認
- ピークの位置と数

〇 左側に偏った分布（右側に長い尾）

例：収入分布

〇 右側に偏った分布（左側に長い尾）

例：試験の点数分布

〇 一様分布

- 全ての値が等しい頻度
- 平坦な形状

演習　セル８～１０による演習

セル8: 基本統計量の算出

【授業知識】PythonとExcelでの統計処理の違い

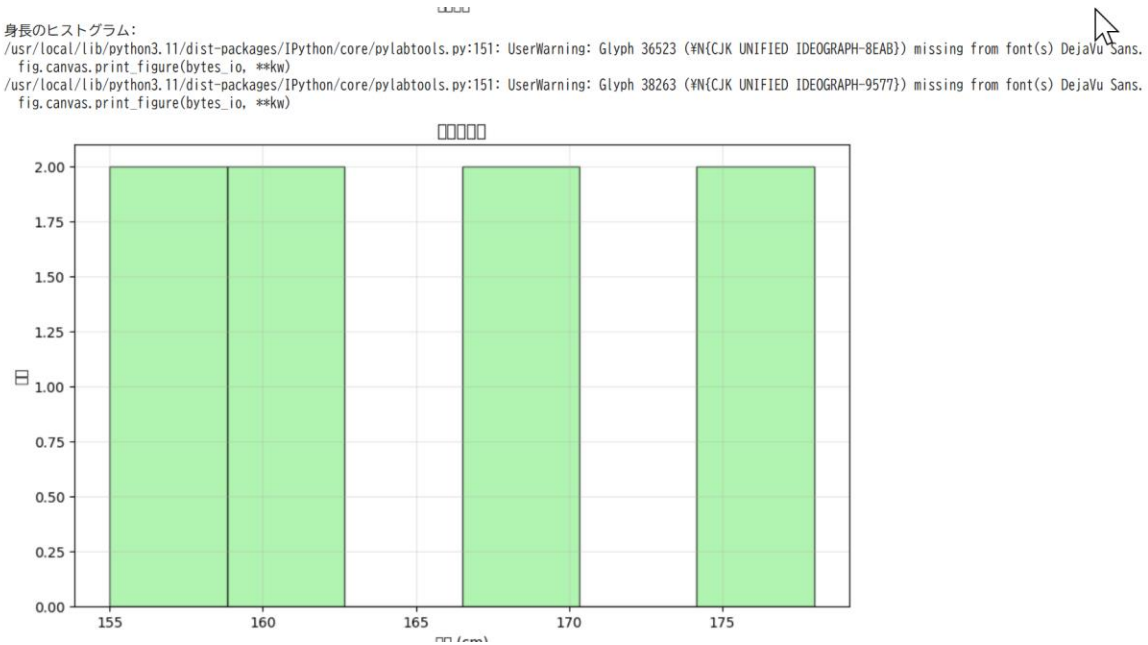
- Excelは少量データの統計処理に適している（視覚的操作，直感的理解）
- Pythonは大量データの処理に優れている（自動化，高速処理，再現性）
- 同じ統計量でも，データ量が増加するとPythonの処理速度の優位性が顕著になる
- プログラムによる処理は再現性があり，同じ処理を繰り返し実行できる
- データサイエンスの現場では両者を適切に使い分けることが重要である

```
=== セル8: 基本統計量の算出 ===
数学成績の基本統計量:
平均値: 83.00
中央値: 83.50
最大値: 95.0
最小値: 72.0

全数値列の統計サマリー:
              年齢          身長         体重        学年         数学         英語
count   7.000000   8.000000   6.000000  8.000000  6.000000  6.000000
mean   20.285714 165.750000  60.000000  2.250000 83.000000 83.833333
std     1.112697   8.293715  11.471704  1.035098  8.294577  5.492419
min    19.000000 155.000000  48.000000  1.000000 72.000000 78.000000
25%    19.500000 159.500000  50.500000  1.750000 77.500000 79.000000
50%    20.000000 165.000000  58.500000  2.000000 83.500000 83.500000
75%    21.000000 171.250000  68.750000  3.000000 87.250000 88.000000
max    22.000000 178.000000  75.000000  4.000000 95.000000 91.000000

              物理
count   7.000000
mean   84.857143
std     5.984106
min    76.000000
25%    81.000000
50%    85.000000
75%    89.500000
max    92.000000
```

セル9: ヒストグラムの作成

【授業知識】データ可視化の重要性

- 数値だけでは見づらいデータの特徴をグラフによって視覚的に理解できる
- ヒストグラムは数値データの分布（散らばり具合）を把握する基本的な手法である
- データの傾向，特異な値の存在，データの偏りなどを直感的に理解できる
- 可視化はデータ分析の結果を他者に説明する際にも重要な役割を果たす



セル10: ヒストグラムの基本的解釈

【授業知識】ヒストグラムから読み取れる情報

- データの散らばり具合：幅広く分布しているか，特定の範囲に集中しているか
- 最頻値付近の確認：最も多くのデータが集まっている値の範囲
- 極端な値の存在：他のデータから大きく離れた値があるかどうか
- 分布の形状：正規分布に近いか，偏りがあるか
- これらの情報は，データの性質を理解し，適切な分析手法を選択するために重要である

```
統計的な確認:
範囲: 72.0点 - 95.0点
平均: 83.0点
中央値: 83.5点
```

ここまでのまとめ

技術的スキル

- Python を用いた基本統計量の算出
- ヒストグラムから、分布形状を読み取る
- 外れ値の発見

確かな基礎

- ExcelとPythonの使い分け
- 基本統計量の意味
- 分布を見ることの意味と重要性

8-4 データ倫理

学習目標

- データ分析やAI開発において倫理的配慮が必要な理由を理解
- AIエンジニアとしての責任と社会的役割を理解できる

学習内容

- データ倫理の基本概念
- AIシステムの社会的影響
- プライバシー保護の重要性
- 公平性とバイアスの問題
- 透明性と説明可能性

データ倫理

データ倫理の主要な原則

- 個人情報の適切な取り扱い
- 最小限必要なデータのみ収集
- 偏見や差別の排除、多様性の尊重

AIエンジニアとしての規範

- 動作原理の説明
- データ利用目的の明確
- AIの判断根拠の説明
- 利用者が理解できる形での情報提供
- 誤りの修正可能性の確保

AIシステムと社会的バイアス

バイアスが生まれる原因

<u>データ収集段階</u>

- 収集方法の偏り

<u>データ処理段階</u>

- 前処理での重要情報の除去
- 特徴量選択の偏り
- ラベル付けの主観性

その他、人間の無意識のバイアス

<u>具体的な事例と影響</u>

問題：過去の採用データで学習したAIが，過去の失敗（採用における差別）を継続

問題：現職や履歴による不当な評価

問題：特定集団のデータ過多や不足による精度の低下

プライバシー保護と個人情報

個人情報の分類

- **直接識別情報**

氏名，住所，電話番号
社会保障番号，マイナンバー

厳重な管理が必要

- **間接識別情報**

年齢・性別・職業の組み合わせ、位置情報，行動パターン

- **機微情報**

健康情報，病歴
信条，思想，政治的意見

特に慎重な取り扱いが必要

プライバシー保護技術

- 匿名化、 仮名化
- 統計的ノイズの追加

個人情報保護法（日本）

- 取得時の利用目的明示
- 本人同意の原則

データ最小化原則

（目的に必要最小限のデータのみ収集、保存期間の適切な設定）

利用者への分かりやすい説明

AIエンジニアの社会的責任

AIエンジニアが持つべき意識

社会的影響の認識
- 技術が社会に与える広範囲な影響
- 意図しない結果への配慮
- 長期的な社会変化への責任

多様性への配慮
- 異なる背景を持つユーザーの考慮
- 文化的差異の尊重
- アクセシビリティの確保

社会動向への敏感さ

演習

セル11: データ倫理とまとめ

【授業知識】データ倫理の重要性

- AIシステムは学習データの偏りをそのまま学習し，社会的偏見を増幅させる可能性がある
- データ分析の結果は判断に大きな影響を与えるため，倫理的配慮が不可欠である
- プライバシー保護，公平性の確保，透明性の維持が基本原則である
- 技術的スキルと同様に，倫理的な判断力がAIエンジニアには求められる
- データを扱う責任と，社会に与える影響を常に意識することが重要である

```
=== セル11: データ倫理とまとめ ===
専攻別の数学平均成績:
専攻
工学    89.333333
文学    76.000000
理学    77.000000
Name: 数学, dtype: float64
```

ここまでのまとめ

理解した概念

- データ倫理の基本原則
- AIシステムにおけるバイアスの問題
- プライバシー保護の重要性
- 社会的公平性の確保
- 透明性と説明可能性の必要性
- AIエンジニアの社会的責任

全体総括：90分で学んだこと

AIエンジニアとしてデータを扱うための実践的な基礎知識とスキルを習得

セクション1

- CSV読み込み、
- 列選択とデータ絞り込み

セクション2

- 欠損値の確認と削除処理、カテゴリ変数の数値変換

セクション3

- 基本統計量の算出、ExcelとPythonの使い分け、データ分布の種類

セクション4

- 倫理的配慮の重要性、社会的責任の認識、公平性とプライバシー保護

最終メッセージ

持ち続けてほしい姿勢

- 好奇心：新しい技術への探究心
- 実践志向：手を動かして学ぶ姿勢
- 論理的思考：データに基づく判断
- 倫理観：社会への責任意識
- 継続力：粘り強い学習習慣